METHOD AND APPARATUS FOR ALLOCATING PROCESSOR BANDWIDTH IN HEAVY-USAGE ENVIRONMENT

ABSTRACT

The method includes detecting and prioritizing one or more interrupt service requests; inserting interrupt servicing instructions responsive to the interrupt service request into an instruction queue mechanism; and processing the instructions within the instruction queue mechanism including the inserted interrupt servicing instructions. The instruction queue mechanism may include an instruction cache and an instruction fetch unit for fetching instructions from the instruction cache, wherein the processing includes decoding the instructions into micro-opcodes and executing the micro-opcodes in one or more out-of-order execution units. The method further includes retiring the executed micro-opcodes including those micro-opcodes representing the inserted interrupt servicing instructions to the instruction cache. Preferably, the criteria for interrupting the core processor include the priority of the interrupts and the capacity of the processor to allocate bandwidth to interrupt servicing. Most preferably, the prioritizing is dynamically responsive to changing allocation criteria, e.g. a current-usage model. In accordance with one embodiment, an interrupt processor determines whether the detected interrupt service request is of a priority meeting one or more defined high-priority criteria and if so then signals the core processor to perform the inserting. Alternatively, the interrupt processor determines whether a natural core processor context switch is imminent and if so then signals the interrupt processor to make ready the highest priority interrupt service request and signals the instruction queue mechanism to fetch the readied interrupt service request in advance of the context switch. The apparatus takes the form of a digital processor for use in a computer supporting one or more hardware interrupt inputs. The processor includes an instruction cache and a fetch-and-decode unit, the fetch-and-decode unit fetching instructions from the instruction cache and decoding the instructions into micro-opcodes. The processor also includes a dispatch-and-execute unit having one or more execution ports, the dispatch-and-execute unit scheduling and executing the micro-opcodes in the one or more execution ports and thereafter retiring the micro-opcodes back into the instruction cache. Finally, the processor includes an interrupt-handling mechanism responsive to one or more

hardware interrupt inputs, the interrupt-handling mechanism instructing the fetch-and-decode unit to insert into a normal instruction sequence decoded micro-opcodes representing interrupt servicing instructions for scheduling and execution by the dispatch-and-execute unit.